CSC 261/461
Database Systems
09/05

Data Models

Data Model:

- A set of concepts to describe the ***structure*** of a database, the ***operations*** for manipulating these structures, and certain ***constraints*** that the database should obey.

Data Model Structure and Constraints:

- Constructs are used to define the database structure
- Constructs typically include ***elements*** (and their ***data types***) as well as groups of elements (e.g. ***entity, record, table***), and ***relationships*** among such groups
- Constraints specify some restrictions on valid data; these constraints must be enforced at all times

Data Models (continued)

Data Model Operations:

- operations used for specifying database *retrievals* and *updates* by referring to the constructs of the data model.

- Operations on the data model may include ***basic model operations*** (e.g. generic insert, delete, update) and ***user-defined operations*** (e.g. *compute_student_gpa*, *update_inventory*)

Categories of Data Models

Conceptual (high-level, semantic) data models:

- Provide concepts that are close to the way many users perceive data.

Physical (low-level, internal) data models:

- Provide concepts that describe details of how data is stored in the computer.

Implementation (representational) data models:

- Provide concepts that fall between the above two, used by many commercial DBMS implementations.

Self-Describing Data Models:

- Combine the description of data with the data values. Examples include XML, key-value stores and some NOSQL systems.

Schemas vs Instances

Database Schema:

- The ***description*** of a database.

- Includes descriptions of the database *structure*, data *types*, and *constraints*.

Schema Diagram:

- An ***illustrative*** display of (most aspects of) a database schema.

Schema Construct:

- A ***component*** of the schema or an object within the schema, e.g., STUDENT, COURSE.

Schemas vs Instances

Database State:

- The actual data stored in a database at a ***particular moment in time***.

- Also called database **instance** (or occurrence or snapshot).

    - The term *instance* is also applied to individual database components, e.g. *record instance*, *table instance*, *entity instance*

Database Schema vs. Database State

Database State:

- Refers to the ***content*** of a database at a moment in time.

Initial Database State:

- Refers to the database state when it is initially loaded into the system.

Valid State:

- A state that satisfies the structure and constraints of the database.

Database Schema vs. Database State (continued)

Distinction

- The ***database schema*** changes very infrequently.
- The ***database state*** changes every time the database is updated.

Schema is also called **intension**.

State is also called **extension**.

Example of a Database Schema

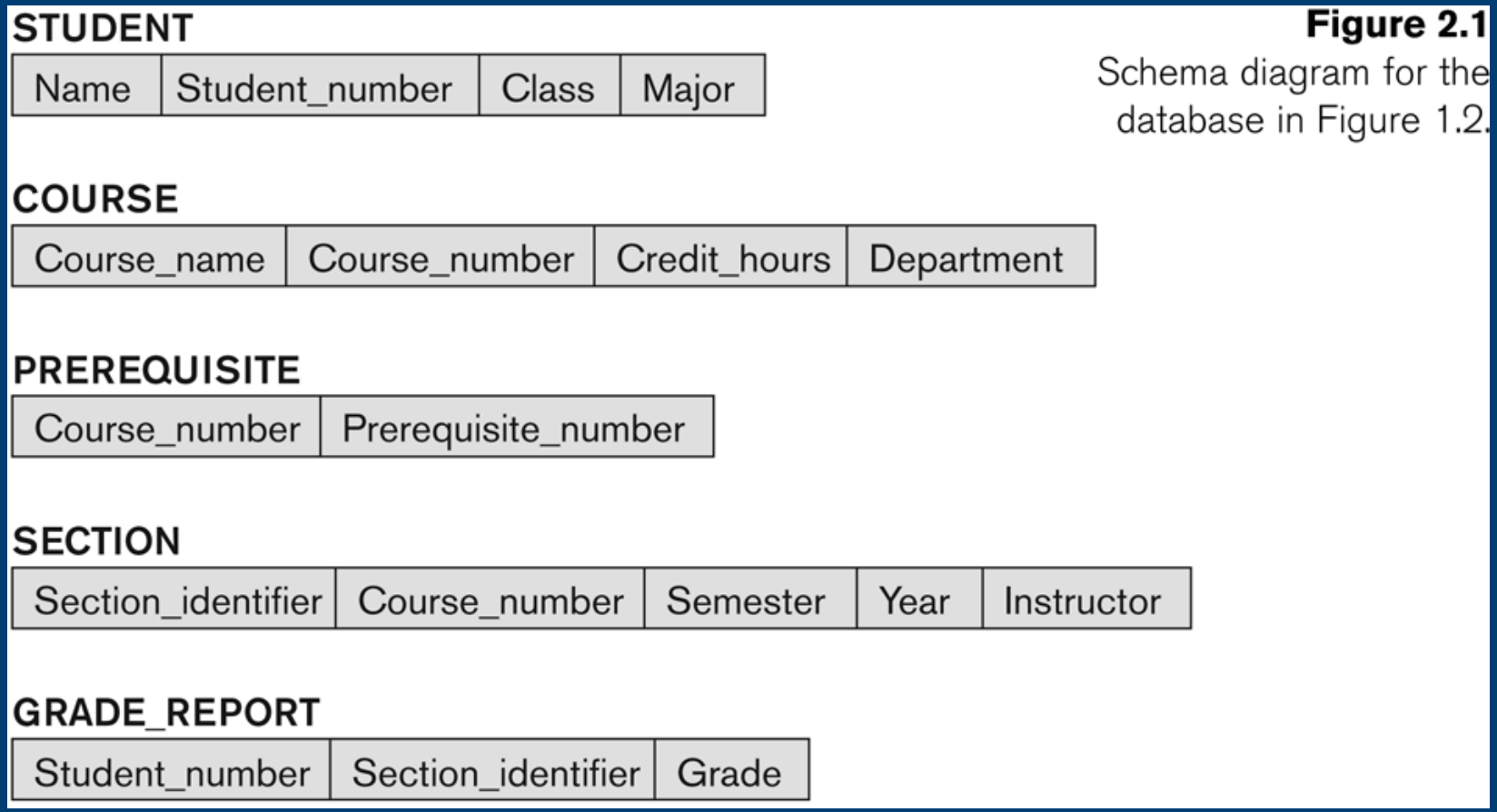


Figure 2.1
Schema diagram for the database in Figure 1.2.

Example of a database state

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

Relational Model Concepts

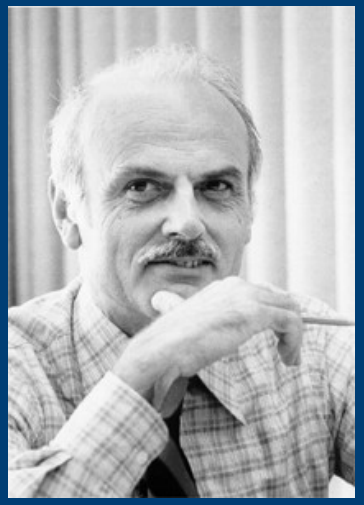
- A **relation** is a mathematical concept based on the ideas of sets

- First proposed by Dr. E.F. Codd of IBM Research:

"A Relational Model for Large Shared Data Banks,"

Communications of the ACM, June 1970

- A major revolution in the field

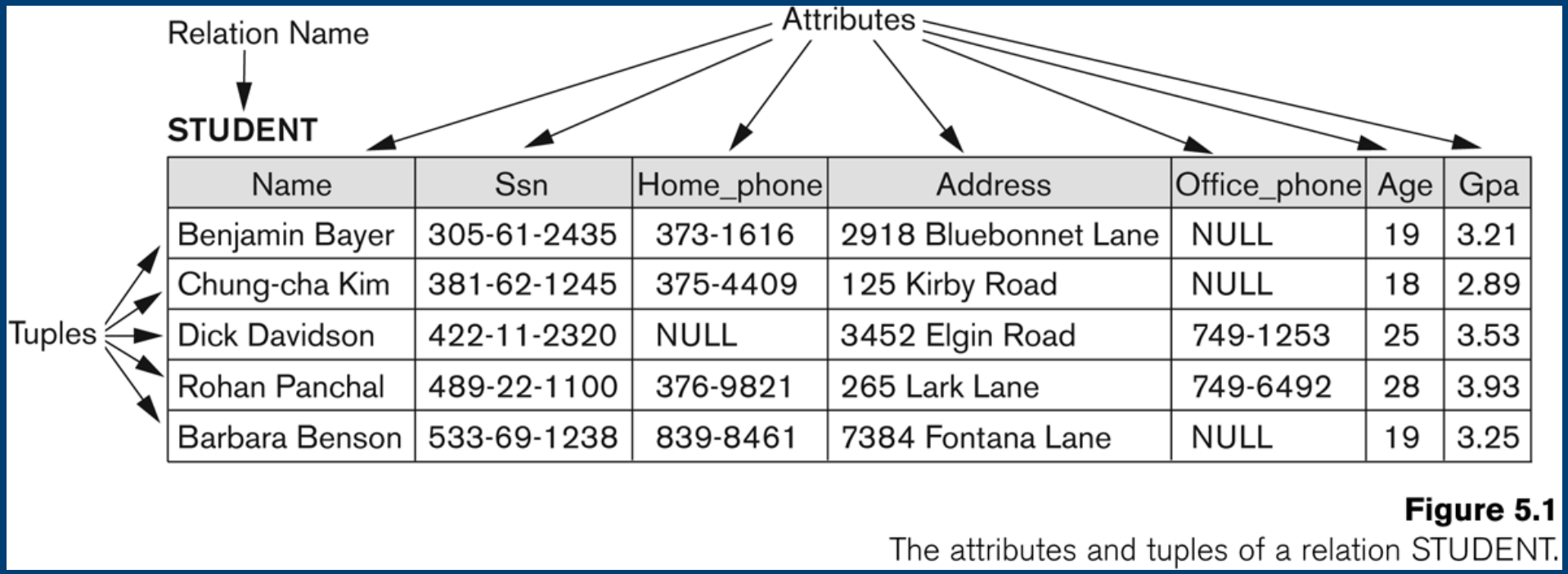


Wikipedia

Definitions

- A **relation** looks like a **table** of values.
- A relation typically contains a **set of rows**.
- Data elements in each **row** represent certain facts that correspond to a real-world **entity** or **relationship**
- In the formal model, rows are called **tuples**
- Each **column** has a column header that gives an indication of the meaning of the data items in that column
- In the formal model, the column header is called an **attribute name** (or just **attribute**)

Example of a Relation



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	749-1253	25	3.53
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	NULL	19	3.25

Figure 5.1
The attributes and tuples of a relation STUDENT.

Definitions

Key of a Relation:

- the value of a data item (or set of items) that uniquely identifies that row in the table

  - What is a *key* for STUDENT?

- Sometimes row-ids or certain numbers are assigned as keys to identify the rows in a table

  - Called *artificial key* or *surrogate key*

Formal Definitions - Schema

Schema (or description) of a Relation:

- Denoted by R(A1, A2, ...An)
- R is the **name** of the relation
- The **attributes** of the relation are A1, A2, ..., An

Example:

`CUSTOMER(Cust-id, Cust-name, Address,Phone#)`

- `CUSTOMER` is the relation name
- Defined over the four attributes: `Cust-id, Cust-name, Address, Phone#`
- Each attribute has a **domain** or a set of valid values.
- For example, the domain of Cust-id is 6 digit numbers.

Definitions - Tuple

A **tuple** is an ordered set of values

- <u, v, x, z>
- Each value is derived from an appropriate *domain*.
  - A row in the CUSTOMER relation is a 4-tuple and would consist of four values, for example:

    <632895, "John Smith", "101 Main St. Atlanta, GA 30332", "(404) 894-2000">
  - This is called a 4-tuple as it has 4 values
- A relation is a **set** of such tuples (rows)

Definitions - State

The **relation state** is a subset of the Cartesian product of the domains of its attributes

- each domain contains the set of all possible values the attribute can take.

Example: attribute Cust-name is defined over the domain of character strings of maximum length 25

The role these strings play in the CUSTOMER relation is that of the *name of a customer*.

Formal Definitions - Summary

Formally,

Given R(A1, A2, ..., An)

r(R) ⊂ dom (A1) X dom (A2) X ....X dom(An)

R(A1, A2, …, An) is the **schema** of the relation

R is the **name** of the relation

A1, A2, …, An are the **attributes** of the relation

r(R): a specific **state** (or "value" or “population”) of relation R – this is a *set of tuples* (rows)

r(R) = {t1, t2, …, tn} where each ti is an n-tuple

ti = <v1, v2, …, vn> where each vj *element-of* dom(Aj)

Questions?